AD NUMBER

AD913119

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Administrative/Operational Use; 17 MAR 1971. Other requests shall be referred to Electronic Systems Division, Hanscom AFB, MA 01731-5700.

AUTHORITY

ESD ltr dtd 4 Dec 1973

Project Report

PA-229-3
(RSP)

Data Reduction Program Documentation
ALREAD

(Effective: March 1971)

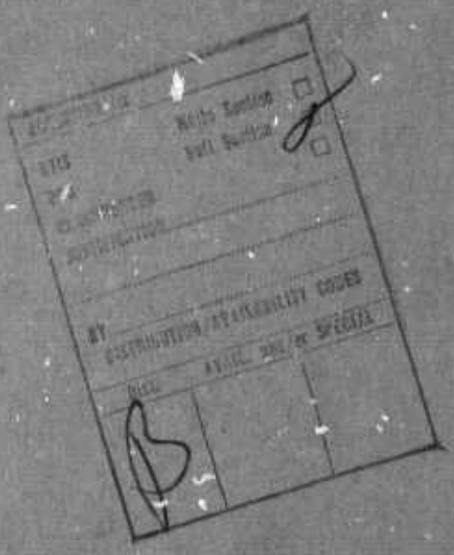
C. R. Berndtson
R. H. French
D. E. Nessman

19621

17 March 1971

Prepared for the Advanced Research Projects Agency, the Department of the Army, and the Department of the Air Force under Electronic Systems Division Contract F19628-70-C-0230 by

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Lexington, Massachusetts



MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

DATA REDUCTION PROGRAM DOCUMENTATION ALREAD (EFFECTIVE: MARCH 1971).

C. R. BERNDTSON,

R. H. FRENCH

D. E. NESSMAN

Philco-Ford Corporation

Editors

43 p.

PROJECT REPORT, PA-229-3 (RSP)

17 MAR 1971

F19628-70-C-0230,

ARPA Order-600

LEXINGTON MASSACHUSETTS

J. G. Harrison Field
Bedford, Mass 01730

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by the Advanced Research Projects Agency of the Department of Defense (ARPA Order 600), the Department of the Army, and the Department of the Air Force under Air Force Contract F19628-70-C-0230.

FOREWORD

This is the third report in the Data Reduction Program Documentation series. It is dated according to the date of completion of the documentation. No implication is made that this program will not subsequently be modified, amended, or superseded: on the contrary, the history of radar data processing is one of continuous evolution of techniques, and it is unrealistic to assume that steady-state has been reached. The PA-229 series is being published for the convenience of interested parties, and Lincoln assumes no responsibility for the correctness of the information presented, nor for its currency.

The preparation of reports in this series is under the Editorship of Charles R. Berndtson of Lincoln, and of D. Nessman and R. French of Philco-Ford Corporation. Inquiries, suggestions, corrections, criticisms, and requests for additional copies should be directed to C. R. Berndtson.

The principal contributor to this report was J. R. Cornelius (Philco-Ford). Due to the intricate, evolutionary manner in which the programs came into being, the editors regret that it is in general impossible to give due credit to all -- mathematicians or radar analysts or programmers -- who contributed to the definition and writing of the programs.

Alan A. Grometstein

CONTENTS

CONTENTS (cont'd)

ALREAD

I. PURPOSE AND UTILIZATION

A. Source of Data

ALTAIR[1]

B. Data Input

ALTAIR Transcription Tape

C. Description

The ALTAIR transcription tape is written in a variable record length format. Subroutine ALREAD is designed to process transcription tape records and extract data necessary to run several programs. It checks for errors, performs computations, and stores the values for the calling program.

ALREAD is called by the following programs: ALTOAK (PA-229-2); ALTAP (PA-229-4); ALT10 (PA-229-5); ALTFENCE (PA-229-6); and ALTMOP (PA-229-). This report applies to these five programs, except for the program listing given in Appendix A. This appendix applies to ALTAP, ALT10, and ALTFENCE. The reports for ALTOAK and ALTMOP contain listings for the version of ALREAD used by each.

D. Output

1. A listing of error messages.
2. A listing of format and calibration records found by Subroutine THEAD and stored in common.
3. Additional output. See calling programs.

II. DESCRIPTION

ALREAD processes data from the ALTAIR transcription tape, one physical record at a time. The data is retrieved from the tape by a call to BREAD, an assembler language routine which reads the tape at a rate of one record per call.[2] ALREAD also calls THEAD, a Fortran routine which, through BREAD, reads the calibration records from the tape, initializes data, and sets up required format tables. ALREAD will return T, Alt, R,* CADJ, Az** and El** for a maximum of 300 consecutive pulses. It also returns RCS data for each pulse and gate.

A call is made to THEAD and data initialization takes place. Vital format tables are tested and, if any are missing, an error message is printed and a return to the calling program made. If all tables are found, the lift-off time† is unpacked and stored. Successive calls to BREAD are made until the record containing the start time is found. This record is unpacked and vital data are tested and stored for use in computations of T, CADJ, RCS, or for direct passage to the calling program.

Data gathering begins at the first pulse whose time ≥ the start time. It stops at the first pulse whose time ≥ the stop time, or when data for the maximum number of pulses has been gathered, whichever occurs first. If a maximum of points are collected before TPLS ≥ the stop time, a code indicates to the calling program that the stop time condition is unsatisfied. The calling program will process this data before recalling ALREAD.

---

*$R = R_{t_0} + \dot{R}_{t_0} (t - t_0)$

where R_{t_0} and $\dot{R}_{t_0}$ are R and $\dot{R}$ at the first pulse in the minor cycle

t is the time of the pulse

t_0 is the time of the first pulse in the minor cycle

**The value closest to t in the major cycle containing the pulse is used.

†If lift-off time is not present, it is set equal to zero.

Pulse times (TPLS) are computed by the formula:

$$TPLS = T_{REC} + (Pulse_n - 1)/PRF$$

where

T_{REC} is the time of the first pulse in the major cycle

$Pulse_n$ is the nth pulse in the major cycle.

RCS is computed as follows:

$$RCS = Amplitude^* + CADJ$$

where

$$CADJ = Calibration\ constant^{**} - Power - 10 * (sensitivity\ bit)^{\dagger} + attenuation + 40 * (\log R)$$

Note that Power and R are used in counts in this computation.

*Amplitude is obtained by indexing the appropriate calibration record (FMAMP1 to FMAMP6)[1] with the amplitude values obtained in FMTRSP, Item 1.[1]

**Found in Calibration Record FMXSEC.[1]

†Found in FMTRAMA, Item 6.[1]

III. OPERATION

A. Input

Call statements for ALREAD, since they vary from program to program, are given in the programs which utilize it.

Definitions of input parameters:

TSTART	Start time of processing (GMT total seconds)
TSTOP	End time of processing (GMT total seconds)
INTARG	Target no. to be processed
INPAT*	Sampling pattern in which initial gate is located
IPOL	Data channel: 1 = LC; 2 = RC; 3 = Az error; 4 = El error
NOPHA	Type of data: 1 = RCS only; 2 = Phase and RCS
NRG	Number of range gates to be processed
ISTGAT**	Location within INPAT of initial gate

B. Input and Output Parameters

NPTS	Output: Number of pulses of data returned
	Input: Must be initialized by calling program before each call to ALREAD†
NEWPAS ††	Cycle and error pointer (see error returns and switch settings)

C. Output

1. Output Parameters

TLIFT	Lift-off time (GMT total seconds)

*Also called IPAT.

**Also called ISG or ING.

†Set to zero for first call. Set to number of saved points for all subsecuent calls.

†† Also called IAGAIN.

DFPG	Frequency and Polarization (e.g. VHF LC)
NFP	Frequency code: 1 = VHF; 2 = UHF

Stored in Common

TIMES	Pulse times (GMT total seconds)
XSPHA	RCS and Phase for each pulse and gate
RANGKM	R
ALSAV*	Alt
ELSAV	El (rad)
IRGA	Range gate array associated with XSPHA
NFPG	Frequency code: 1 = VHF; 2 = UHF
CADJ**	Adjusted calibration constant
AZI	Az (rad)
ELE	El (rad)
FSIG	$RCS^{\frac{1}{2}}$ (m) for each pulse and gate
IPHASE	Phase for each pulse and gate
DOPM	Range rate (km/s)
PRD	PRF
IPOL	Data channels wanted
NPOL	Number of data channels used

2. Error Returns and/or Switch Settings

NEWPAS = 0 Indicates that re-entry to ALREAD is not desirable either because TSTOP has been reached or because a fatal error has occurred. (See below)

a. "THEAD has defaulted - run aborted"
b. "At time = (12345.6789) Target (00) is not on the tape, targets available are (00, 00, 00, 00, 00)"
c. "Either Polarization or pattern chosen is not available"

*Valid only for first pulse of minor cycle. It is repeated for subsequent pulses.
**Also called CALOUT.

NEWPAS = 1	Indicates a tape has been rewound.
NEWPAS = 44	Indicates that an ALTAIR data end-of-file record has been read from the tape. A message is printed by ALREAD. "End-of-file encountered on input tape after time (12345.6789)"
NEWPAS = 55	Indicates that a vital format table is missing; processing must stop. A message identifying the table is printed. "Format table (NAME) was not found - run aborted"
NEWPAS = 66	Indicates that an amplitude or phase table for a requested polarization is missing. A message identifying the table is printed.
NEWPAS = 99	Indicates that the maximum number of points per call have been stored but that TSTOP has not been reached.

3. ALREAD-generated error messages and accompanying switch setting.

a. "At time (12345.6789), minor cycle (0), something is screwed up NBP* = (01234). It should be (43210)" NEWPAS is not set. ALREAD will continue by skipping data in current record, and calling BREAD for next record.

b. "NPLS = (00) Minor cycle (0), (0) minor cycles" A test for a value of NPLS ≤ 0. Same action as in a.

c. "At time (12345.6789) R = (12345.678900)" A test for a value of R ≤ 0. If this occurs, R is set to 1.0 and processing continues.

*NBP is number of bits per pulse and is coded on the tape. It is also computed. If the two values are not identical, some pointers are wrong and the data may be faulty.

IV. PROGRAM LIMITATIONS

NRG	≤ 120 (30 for ALTMOP and ALTOAK)
NPTS per call	≤ 300 for RCS data (NOPHA = 1) ≤ 150 for RCS and phase data (NOPHA = 2)

V. PROGRAMMING

A. ALREAD

A program listing and a flow diagram for ALREAD are given in Appendices A and B.

ALREAD calls the following subroutines:

B. THEAD (see Appendices C and D).

THEAD is used to process calibration and format records which are recorded before the data records on the ALTAIR transcription tape. THEAD lists format and calibration records named in the common statement. These are stored and unpacked for later use by the main processing program. A sample THEAD output is given in Appendix E.

The minimum size needed for the item array may be calculated by the following equation:

Item size = 6 * (Total number of items stored) + (6 * 130)

Calling Sequence:

Call THEAD (NEWPAS, *)

NEWPAS = Tape processing status; integer variable.

- 0 = Initial pass thru tape; key calibration and format records stored and unpacked.
- 1 = Tape has been rewound; skip calibration and format records but call BREADS to reinitialize parameters if data record processing is to continue in main program.
- * = A return point specified by a statement number in the calling program. Used for aborting job by main program if wrong tape is mounted.

THEAD calls the following subroutines:

BREADS (entries BREADS and BREAD); HDRR (entries HDRR and NAMET); and FORM.[2] THEAD also calls WHICHV, a 360-system subroutine indicating whether a job is being run under the time-sharing (CMS) or Batch (OS) systems.

C. GMTUPK (see Appendix F)

GMTUPK is a routine to unpack GMT from the tape. The call is GMTUPK (TIME).

GMTUPK input, which is stored in common, is two packed data words, one containing h, min, s, and the other word containing decimal fractions of a second (GMT).

The output (TIME) is GMT total seconds, and is contained in the call statement.

D. GET and IGET

GET and IGET are entries to subroutine GETS.[2]

These routines will locate any data item, unpack it, and interpret it according to the information in the format table. They will return the item as a binary integer (in the case of IGET) or as a floating point number (in the case of GET).

GET (or IGET) requires three arguments:

GET (Format, Base, Item)

Format	Relevant format table address
Base	Base address of data block desired
Item	Specific item number

E. BREAD

An entry to subroutine BREADS used to read data records from the ALTAIR tapes.[2]

REFERENCES

1. "ALTAIR Data User's Manual", LM-97, Lincoln Laboratory, M.I.T. (to be published), UNCLASSIFIED.

2. "Data Reduction Program Documentation, ALTAIR Tape Read Package, (Effective: April 1970)", PA-229-1, Lincoln Laboratory, M.I.T. (17 March 1971), UNCLASSIFIED.

COMMON SYMBOLS AND ABBREVIATIONS

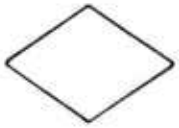
(The units given for certain quantities are the units commonly used for those quantities, unless otherwise noted.)

ADT	ALCOR Data Tape
Alt	Altitude (km)
APS	Average Pulse Shape
ARS	ALTAIR Recording System
Avg	Average, Averaging
Az	Azimuth (deg)
CADJ	Adjusted Calibration Constant (db)
C-band	ALCOR frequency, 5664 MHz (NB) and 5667 MHz (WB)
El	Elevation (deg)
EOF	End of File
GMT	Greenwich Mean Time
h	Hours
Hz	Hertz
in	Inches
LC	Left Circular Polarization
min	Minutes
NB	Narrow Band
NRTPOD	Non-real Time Precision Orbit Determination Program
POD	Project PRESS Operation and Data Summary Report
Phase	Presented in deg
PRF	Pulse Repetition Frequency (pps)
PRI	Pulse Repetition Interval (s)
pps	Pulses per second
pts	Points

R	Range (km)
$\dot{R}$	Range Rate (km/s)
rad	Radians
RC	Right Circular Polarization
RCS	Radar Cross Section (dbsm)
s	Seconds
SD_w	Standard Deviation of Wake Velocity
T	Time
TAL	Time After Launch (s)
UHF	ALTAIR Frequency; 415 MHz
V	Velocity
V_d	Doppler Velocity
V_w	Mean Wake Velocity
VHF	ALTAIR Frequency; 155.5 MHz
WB	Wide Band
θ	Total Off-axis Angle (deg)
λ	Wavelength
*	Denotes Multiplication

FLOW DIAGRAM SYMBOLS

PROCESS, ANNOTATION



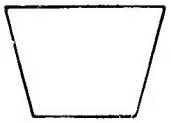
DECISION

TERMINATOR

SUBROUTINE: where NAME is the entry call into the subroutine

CONNECTOR: where P specifies a page in the flow diagram, and L designates a statement number in the program listing or a reference point in the flow diagram

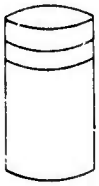
CONNECTOR: where X implies a continuation of the diagram to the next page



INPUT/OUTPUT OPERATION

MAGNETIC TAPE

PUNCHED CARD



DISK

APPENDIX A

SUBROUTINE ALREAD PROGRAM LISTING

```
      SUBROUTINE ALREAD(TSTART,TSTOP,TLIFT,INTARG,INPAT,IPOL,NOPHA,NPTS,
     1DFPG,NEWPAS,NRG,ISTGAT)
      DIMENSION ALT(5),AMT(3,5),AZ(6),CALCON(19),DFPG(2),EL(6),IHD(13),
     1IMT(5),INGATE(5,3),IPRIOR(5),ISENS(6),ISLIDE(5,3),ISPAC(5,3),
     2ITARDT(5,3,3),ITARG(5),IWAVE(6),LOC(5,3,4),MODE(5,3),NEX(2),
     3NMODES(5,3),NPTEST(2),NSAMP(5,3),NSAMPT(5),POL(4),POWER(6),
     4RAD(6,5),RANGE(5),TYPE(2),VEL(5),IFPAR(22)
      COMMCN/TREAD/LN,IELG,IBTRHD,FMTRHD,FMTRMA,FMTRTG,FMTRMI,FMTRSP,
     1FMXSEC,FMARSG,FMRR11,FMGLOT,FMCHAP,FMBSMC,FMASLP,FMAMP(6),FMPHA(6)
     2,NAME(25),NI(24),IX(24),TAMP(128,6),TPH(128,6),ITEM(2000)
      COMMON/TIMCOM/IHMS,IFS
      COMMCN/RDCOMT/TIMES(300),XSPHA(120,300),RANGKM(300),ALSAV(300),
     1IRGA(120),NFPG
      EQUIVALENCE (IHD(1),IDREC),(IHD(2),LREC),(IHD(3),ITGTM1),
     1(IHD(4),ITGTM2),(IHD(5),IFPG),(IHD(6),IPRI),(IHD(7),NBLRD),
     2(IHD(8),MACYBA),(IHD(9),NTARG),(IHD(10),NTDBA),(IHD(11),NMINOR),
     3(IHD(12),MICYBA),(IHD(13),LMICY),(IFPAR(1),FMTRHD),
     4(PAD(1,1),POWER(1)),(RAD(1,2),AZ(1)),(RAD(1,3),EL(1)),
     5(ITARDT(1,1,1),MODE(1,1)),(ITARDT(1,1,2),ISPAC(1,1)),
     6(ITARDT(1,1,3),NSAMP(1,1)),(IMT(1),IGCHG),(IMT(2),IGAIN),
     7(IMT(3),NPLS),(IMT(4),ISDBA),(IMT(5),NBP),(AMT(1,1),RANGE(1)),
     8(AMT(1,2),VEL(1)),(AMT(1,3),ALT(1))
      DATA PCON,RKM,VKM/1.0E6,1.873703E-3,4.4672E-4/
      DATA POI/'LC  RC  AZ  EL  '/
      DATA TYPE/'VHF-UHF-'/
      DATA NEX,NPTEST/0,1,300,150/
      DOUBLE PRECISION DRANG,FINTIM,FPG,GMTIME,PRF,TIME(6),TIMES,TIMOLD,
     1TLIFT,TSTART,TSTOP
      INTEGER*2 ITEM
      IF(NEWPAS.GT.2)GO TO 2000
      IF(NRG.GT.120)NRG=120
      GMTIME=TSTART-1.0
      NPTS=0
      TINC=0.0
      NTERR=0
      CALL THEAD(NEWPAS,89960)
      DO 60 I=1,6
      IF(IFPAR(I).GT.0)GO TO 60
      WRITE(6,40)NAME(I)
40    FORMAT(' FORMAT TABLE ',A4,' WAS NOT FOUND - RUN ABORTED.')
      NEWPAS=55
      RETURN
60    CONTINUE
80    DO 100 I=2,19
      CALCON(I)=GET(FMXSEC,IBTRHD,I)
100   CONTINUE
      TLIFT=0.0
      ITL=IGET(FMGLOT,IBTRHD,1)
      IF(ITL.NE.2)GO TO 140
      IHMS=IGET(FMGLOT,IBTRHD,2)
      IFS=IGET(FMGLOT,IBTRHD,3)
      CALL GMTUPK(TLIFT)
      GO TO 140
120   CALL BREAD(1)
140   IHD(1)=IGET(FMTRHD,IBTRHD,1)
```

```
      IF((IDREC.LT.127).AND.(IFLG.NE.3))GO TO 180
      WRITE(6,160)GMTIME
160   FORMAT(' END OF FILE ENCOUNTERED ON INPUT TAPE AFTER TIME ',F12.4)
      NEWPAS=44
      RETURN
180   IF(IDREC.NE.2)GO TO 120
      TIMOLD=GMTIME
      DO 200 I=2,13
      IHD(I)=IGET(FMTRHD,IBTRHD,I)
200   CONTINUE
      IHMS=IGET(FMTRHD,IBTRHD,14)
      IFS=IGET(FMTRHD,IBTRHD,15)
      CALL GMTUPK(GMTIME)
      IF(GMTIME.LT.(TSTART-0.20))GO TO 120
      IF((GMTIME.GT.TIMOLD).AND.(MACYBA.EQ.0).AND.(NTDBA.EQ.0))GO TO 120
      NCONT=0
      IF(((DABS(TIMOLD-GMTIME)).GT.0.01).AND.(NTDBA.NE.0))GO TO 240
      IF(NTERR.NE.0)GO TO 120
      NCONT=1
      GO TO 1200
240   NCHAN=IFPG+IPOL
      IF(IFPG.EQ.1)NCHAN=NCHAN+3
      NFPG=IFPG+1
      DFPG(1)=TYPE(NFPG)
      DO 280 I=1,NOPHA
      INDEX=(I-1)*6+NCHAN+12
      IF(IFPAR(INDEX).NE.0)GO TO 280
      WRITE(6,40)NAME(INDEX)
      NEWPAS=66
      RETURN
280   CONTINUE
      FINTIM=GMTIME
      TIMOLD=GMTIME
      DO 300 I=1,5
      NSAMPT(I)=0
300   CONTINUE
      PRF=PCON/FLOAT(IPRI)
      TINK=1./PRF
      IF(NBLRD.EQ.0)GO TO 600
      DO 500 I=1,NBLRD
      IBTRMA=IBTRHD+MACYBA+24*(I-1)
      DO 400 K=1,3
      RAD(I,K)=GET(FMTRMA,IBTRMA,K)
400   CONTINUE
      IHMS=IGET(FMTRMA,IBTRMA,4)
      IFS=IGET(FMTRMA,IBTRMA,5)
      CALL GMTUPK(TIME(I))
      ISENS(I)=IGET(FMTRMA,IBTRMA,6)
      IWAVE(I)=IGET(FMTRMA,IBTRMA,7)
      IF(IWAVE(I).EQ.4)IWAVE(I)=3
      IF(IWAVE(I).GT.3)IWAVE(I)=0
500   CONTINUE
      KONGET=3*(NCHAN-1)+IWAVE(1)+1
      CALADD=CALCON(KONGET)-POWER(1)-FLOAT(10*ISENS(1))
600   IF(NTARG.EQ.0)GO TO 120
```

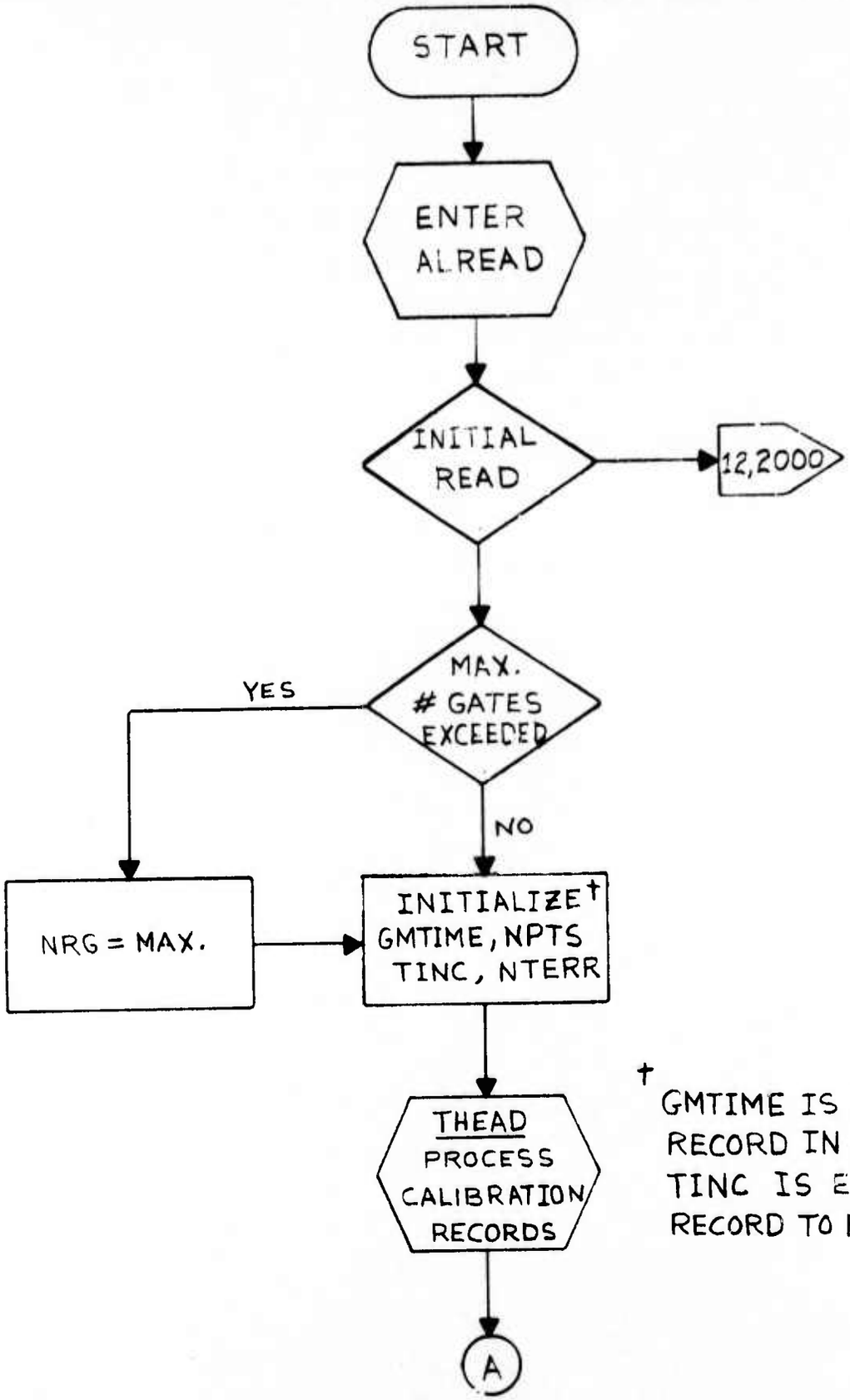
```
      ITPIK=0
      NBPTST=0
      DO 900 I=1,NTARG
      IBTRTG=IBTRHD+NTDBA+12*(I-1)
      ITARG(I)=IGET(FMTRTG,IBTRTG,1)
      IPRIOR(I)=IGET(FMTRTG,IBTRTG,2)
      NK=1
      DO 800 K=1,3
      IK=NK+1
      NK=IK+2
      DO 700 L=1,3
      ITARDT(I,K,L)=IGET(FMTRTG,IBTRTG,L+IK)
700   CONTINUE
      IF(ISPAC(I,K).EQ.254)ISPAC(I,K)=-1
      ISPAC(I,K)=2**(ISPAC(I,K)+NEX(NFPG))
      IF(ISPAC(I,K).EQ.0)ISPAC(I,K)=1
      NSAMPT(I)=NSAMPT(I)+NSAMP(I,K)
      NMODES(I,K)=0
      DO 780 L=1,4
      LOC(I,K,L)=0
      MODUM=MOD(MODE(I,K),2)
      IF(MODUM.EQ.0)GO TO 760
      NMODES(I,K)=NMODES(I,K)+1
      LOC(I,K,L)=NMODES(I,K)
760   MODE(I,K)=MODE(I,K)/2
780   CONTINUE
      ISLIDE(I,K)=2*NSAMP(I,K)*NMODES(I,K)
      NBPTST=NBPTST+ISLIDE(I,K)
800   CONTINUE
      DO 820 J=1,3
      INGATE(I,J)=IGET(FMTRTG,IBTRTG,J+11)
820   CONTINUE
      IF(ITARG(I).EQ.INTARG)ITPIK=I
900   CONTINUE
      NBPTST=NBPTST-MOD(NBPTST,6)
      NTERR=0
      IF(ITPIK.GT.0)GO TO 960
      WRITE(6,920)GMTIME,INTARG,(ITARG(I),I=1,NTARG)
920   FORMAT('  AT TIME = ',F12.4,' TARGET ',I2,' IS NOT ON THE TAPE, TA
     1RGETS AVAILABLE ARE ',5I3)
      IBTRMI=IBTRHD+MICYBA
      NBP=IGET(FMTRMI,IBTRMI,5)
      IF(NBP.EQ.NBPTST)GO TO 10000
      NTERR=1
      GO TO 120
960   IF(LCC(ITPIK,INPAT,IPOL).GT.0)GO TO 1020
      WRITE(6,1000)
1000  FORMAT(' EITHER POLARIZATION OR PATTERN CHOSEN IS NOT AVAILABLE')
      GO TO 10000
1020  IF(ISTGAT.GT.NSAMP(ITPIK,INPAT))ISTGAT=1
      IF(ISTGAT.LT.1)ISTGAT=1
      NSTEST=NSAMP(ITPIK,INPAT)-ISTGAT+1
      NSTGAT=ISTGAT
      IPAT=INPAT
      I=1
```

```
1040  IRGA(I)=INGATE(ITPIK,IPAT)+(NSTGAT-1)*ISPAC(ITPIK,IPAT)
      NSTGAT=1
1060  IF(I.GE.NRG)GO TO 1100
      IF((I+1).GT.NSTEST)GO TO 1080
      I=I+1
      IRGA(I)=IRGA(I-1)+ISPAC(ITPIK,IPAT)
      GO TO 1060
1080  IPAT=IPAT+1
      IF(IPAT.GT.3)GO TO 1100
      IF(NSAMP(ITPIK,IPAT).LE.0)GO TO 1080
      NSTEST=NSTEST+NSAMP(ITPIK,IPAT)
      I=I+1
      IF(I.LE.NRG)GO TO 1040
      I=I-1
1100  IPAT=INPAT
      NRG=I
      DFPG(2)=POL(IPOL)
1200  IBTRMI=IBTRHD+MICYBA
      DO 2200 MIN=1,NMINOR
      IF(MIN.LE.1)GO TO 1220
      IBTRMI=IBTRMI+LMICY+NPLS*NBP
1220  DO 1240 I=1,5
      IMT(I)=IGET(FMTRMI,IBTRMI,I)
1240  CONTINUE
      IF(NBP.EQ.NBPTST)GO TO 1280
      WRITE(6,1260)GMTIME,MIN,NBP,NBPTST
1260  FORMAT(' AT TIME = ',F10.4,', MINOR CYCLE ',I1,' SOMETHING IS SCRE
     1WED UP NBP = ',I5,' IT SHOULD BE ',I4,' - SHOOT HARTOGENSIS')
      GO TO 1330
1280  IF(NPLS.GE.1)GO TO 1340
      WRITE(6,1320)NPLS,MIN,NMINOR
1320  FORMAT(' NPLS = ',I2,3X,'MINOR CYCLE ',I1,3X,I1,' MINOR CYCLES')
1330  GMTIME=GMTIME+TINC
      GO TO 120
1340  TINC=FLOAT(NPLS)/PRF
      DO 1500 L=1,NTARG
      IBTRMT=IBTRMI+(L-1)*12
      DO 1400 I=1,3
      AMT(L,I)=GET(FMTRMI,IBTRMT,I+5)
1400  CONTINUE
      VEL(I)=VEL(L)*VKM
1500  CONTINUE
      IBTRMT=IBTRMI+(ITPIK-1)*12
      ISDREL=IGET(FMTRMI,IBTRMT,9)
      NSDUM=NSAMPT(ITPIK)
      IF(NRG.LT.NSDUM)NSDUM=NRG
      DRANG=RANGE(ITPIK)
      IF(DRANG.GT.0.0)GO TO 1560
      WRITE(6,1540)GMTIME,DRANG
1540  FORMAT('  AT TIME = ',F15.4,' THE RANGE = ',1PE20.6)
      DRANG=1.0
1560  DRANG=40.*DLOG10(DRANG)
      CALIB=DRANG+FLOAT(IGAIN)+CALADD
      DO 2000 I=1,NPLS
      IF((MIN.GT.1).OR.(I.GT.1))FINTIM=FINTIM+TINK
```

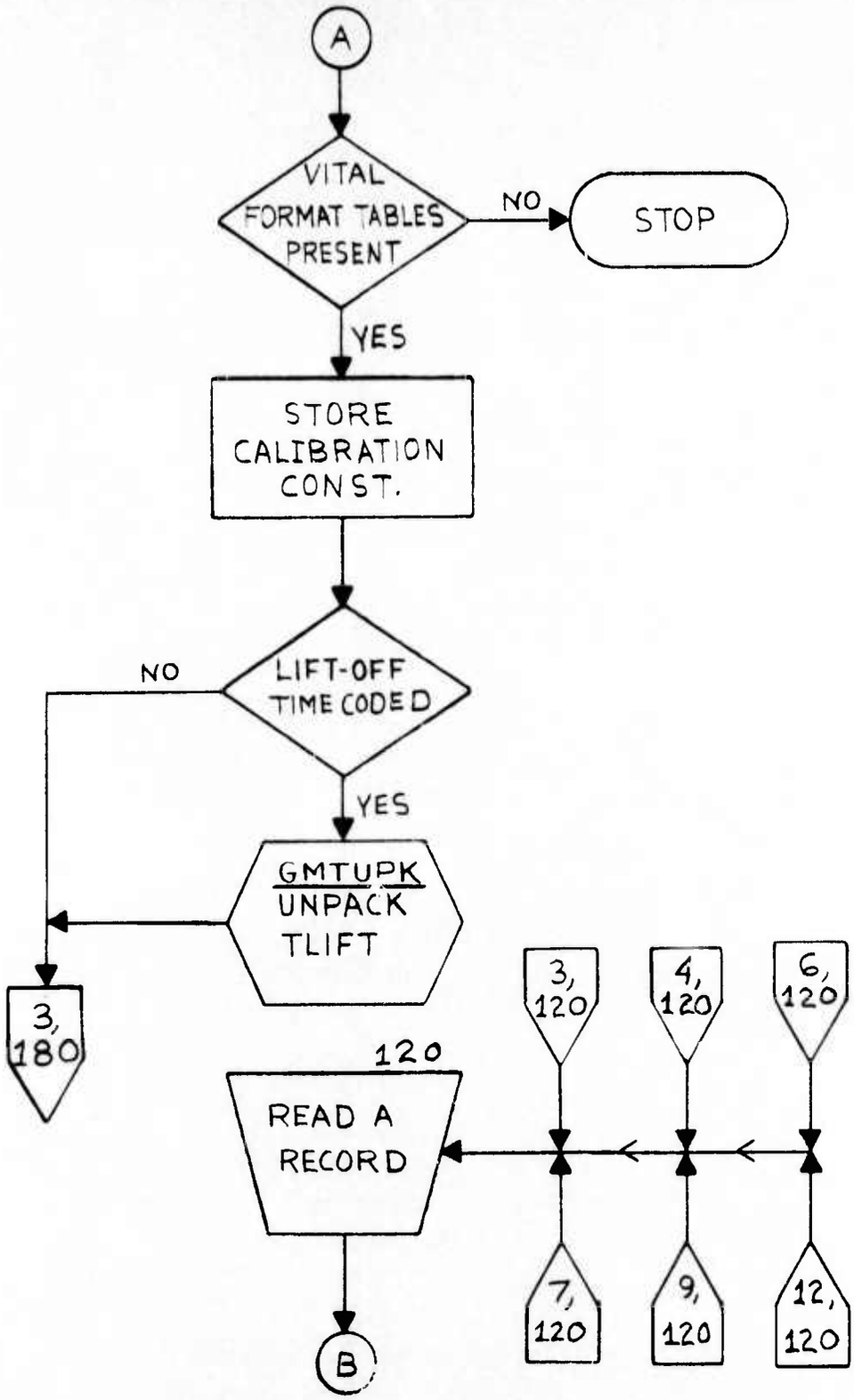
```
      IF(TSTART.GT.FINTIM)GO TO 2000
      NPTS=NPTS+1
      IF(NPTS.GT.1)GO TO 1600
1580  TIMES(NPTS)=FINTIM
      GO TO 1620
1600  IF((I.EQ.1).AND.(MIN.EQ.1).AND.(NCONT.EQ.0))GO TO 1580
      TIMES(NPTS)=TIMES(NPTS-1)+TINK
1620  RANGKM(NPTS)=RANGE(ITPIK)*RKM+VEL(ITPIK)*(TIMES(NPTS)-FINTIM)
      ALSAV(NPTS)=ALT(ITPIK)
      ISAMPT=IBTRHD+ISDBAS+ISDREI*(I-1)*NBP
      NSTEST=NSAMP(ITPIK,INPAT)-ISTGAT+1
      IAD=0
      IF(INPAT.EQ.1)GO TO 1680
      JST=INPAT-1
      DO 1660 J=1,JST
      IAD=IAD+ISLIDE(ITPIK,J)
1660  CONTINUE
1680  L=ISTGAT-1
      DO 1800 K=1,NRG
      L=L+1
      IF(K.LE.NSTEST)GO TO 1700
      IAD=IAD+ISLIDE(ITPIK,IPAT)
      IPAT=IPAT+1
      NSTEST=NSTEST+NSAMP(ITPIK,IPAT)
      L=1
1700  IPIK=ISAMPT+IAD+2*(LOC(ITPIK,IPAT,IPOL)-1+NMODES(ITPIK,IPAT)
     1*(L-1))
      IAMP=IGET(FMTRSP,IPIK,1)
      IF((IAMP.LT.1).OR.(IAMP.GT.128))IAMP=1
1720  XSPHA(K,NPTS)=TAMP(IAMP,NCHAN)+CALIB
1740  IF(NOPHA.EQ.1)GO TO 1800
      IPHA=IGET(FMTRSP,IPIK,2)
      IF((IPHA.GE.0).AND.(IPHA.LE.127))GO TO 1760
      XSPHA(K,NPTS+150)=0.0
      GO TO 1800
1760  XSPHA(K,NPTS+150)=TPH(IPHA+1,NCHAN)
1800  CONTINUE
      IPAT=INPAT
      IF(TSTOP.LT.TIMES(NPTS))GO TO 10000
      IF(NPTS.LT.NPTEST(NOPHA))GO TO 2000
      NEWPAS=99
      RETURN
2000  CONTINUE
2200  CONTINUE
      GO TO 120
9960  WRITE(6,9980)
9980  FORMAT('  THEAD HAS DEFAULTED - RUN ABORTED.')
10000 NEWPAS=0
      RETURN
      END
```

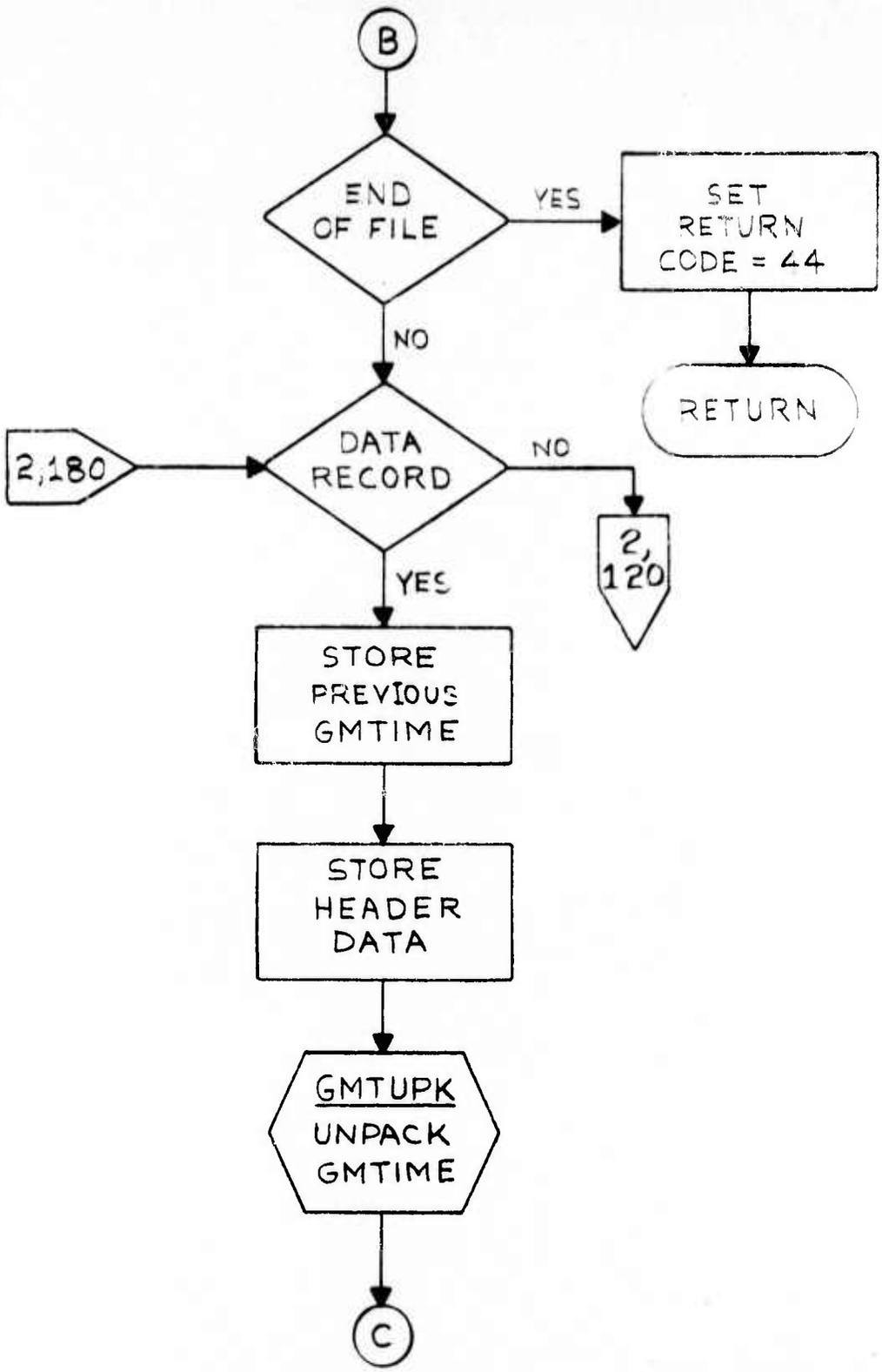
APPENDIX B

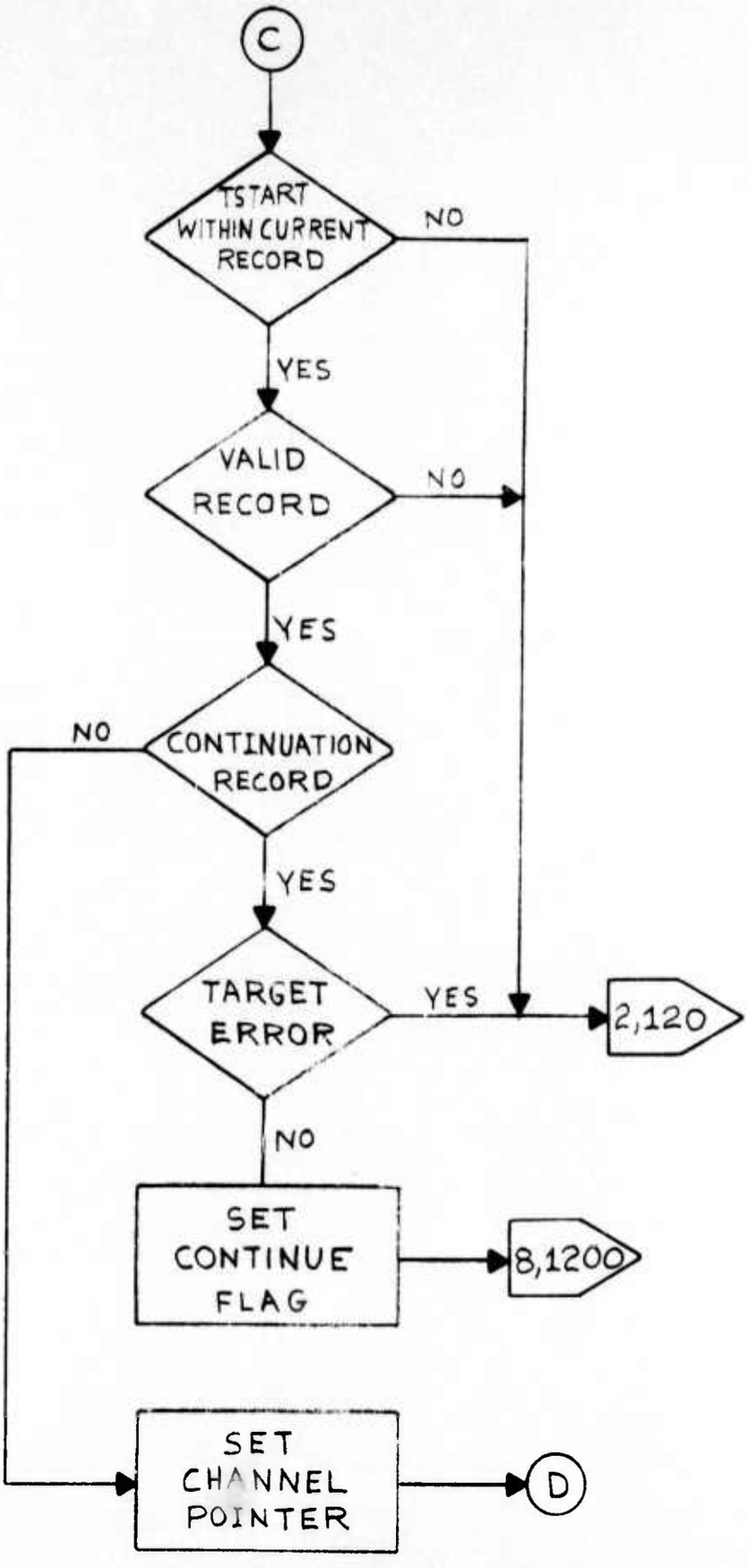
SUBROUTINE ALREAD FLOW DIAGRAM

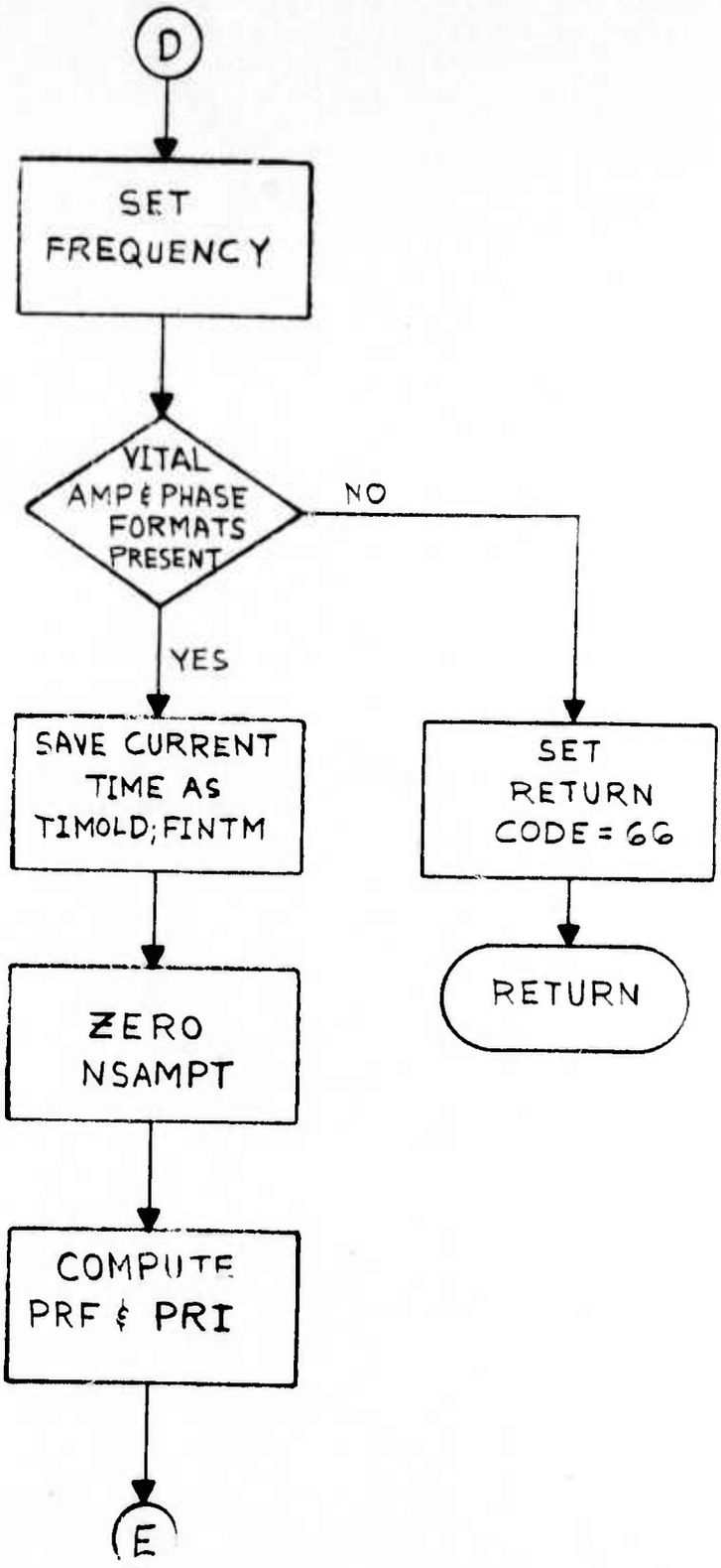


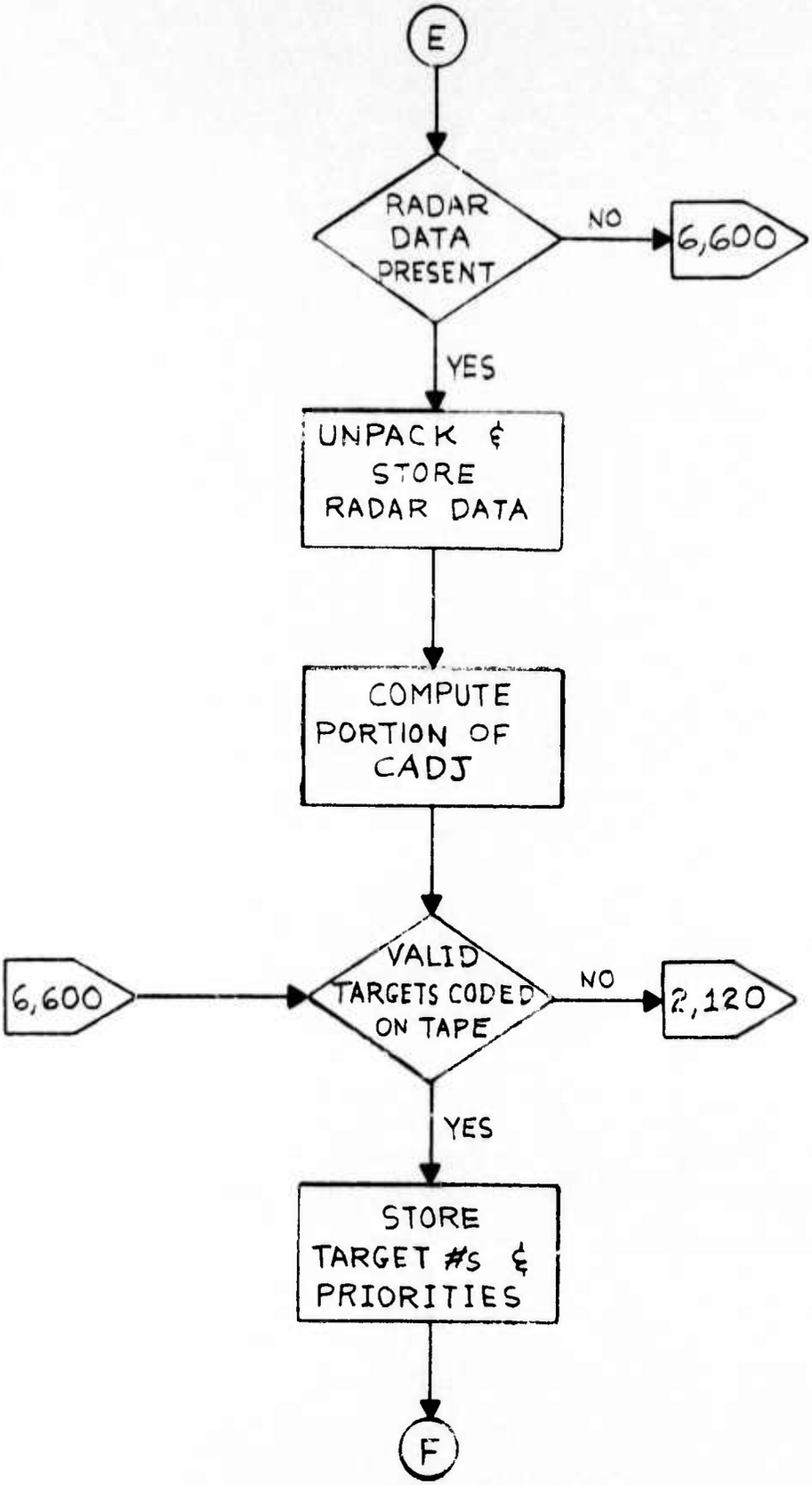
† GMTIME IS TIME OF RECORD IN TOTAL GMT. TINC IS ESTIMATED RECORD TO RECORD TIME.

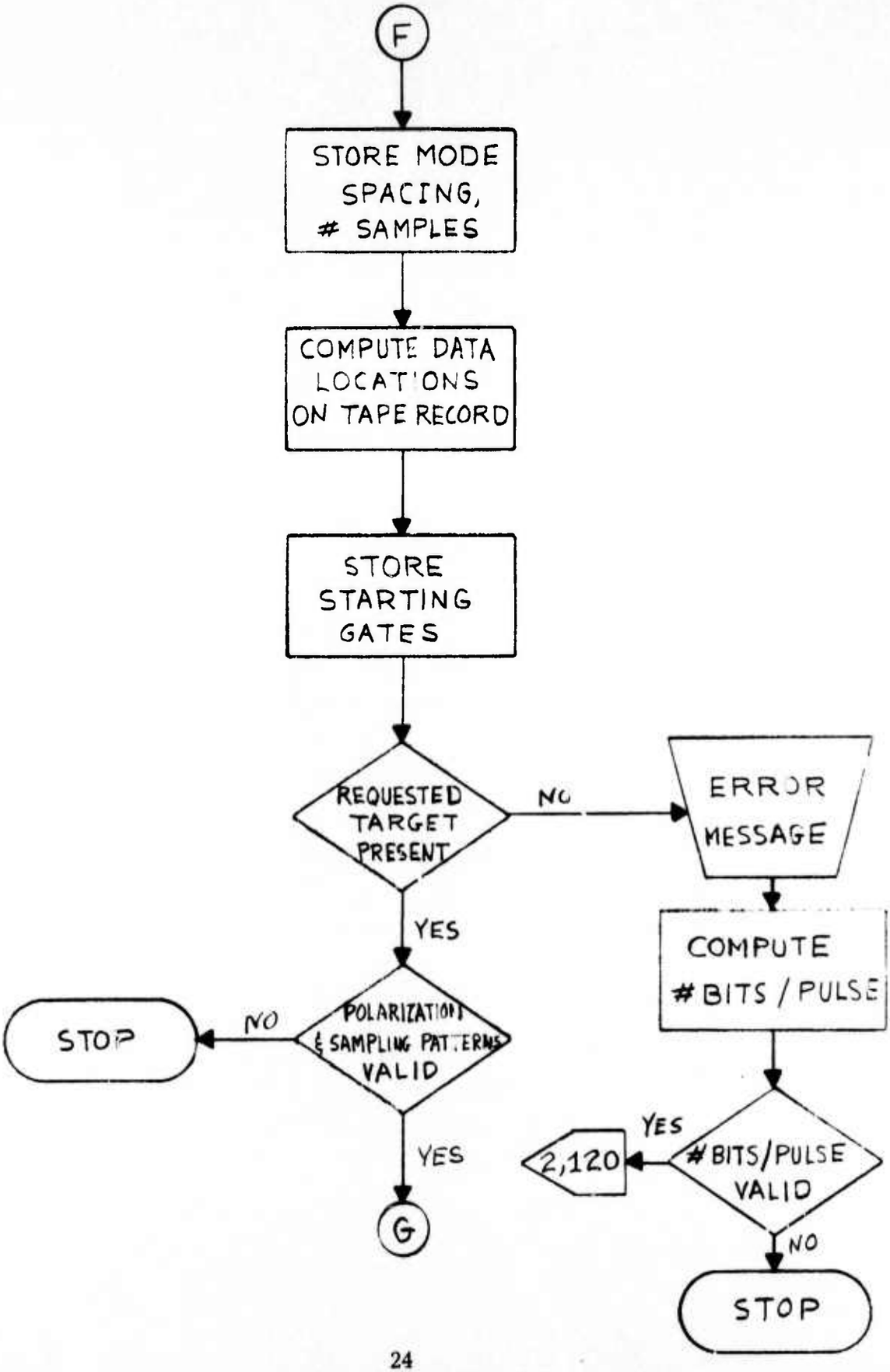

A
VITAL FORMAT TABLES PRESENT
NO
STOP
YES
STORE CALIBRATION CONST.
LIFT-OFF TIME CODED
NO
YES
GMTUPK
UNPACK
TLIFT
3, 180
120
READ A RECORD
3, 120
4, 120
6, 120
7, 120
9, 120
12, 120
B

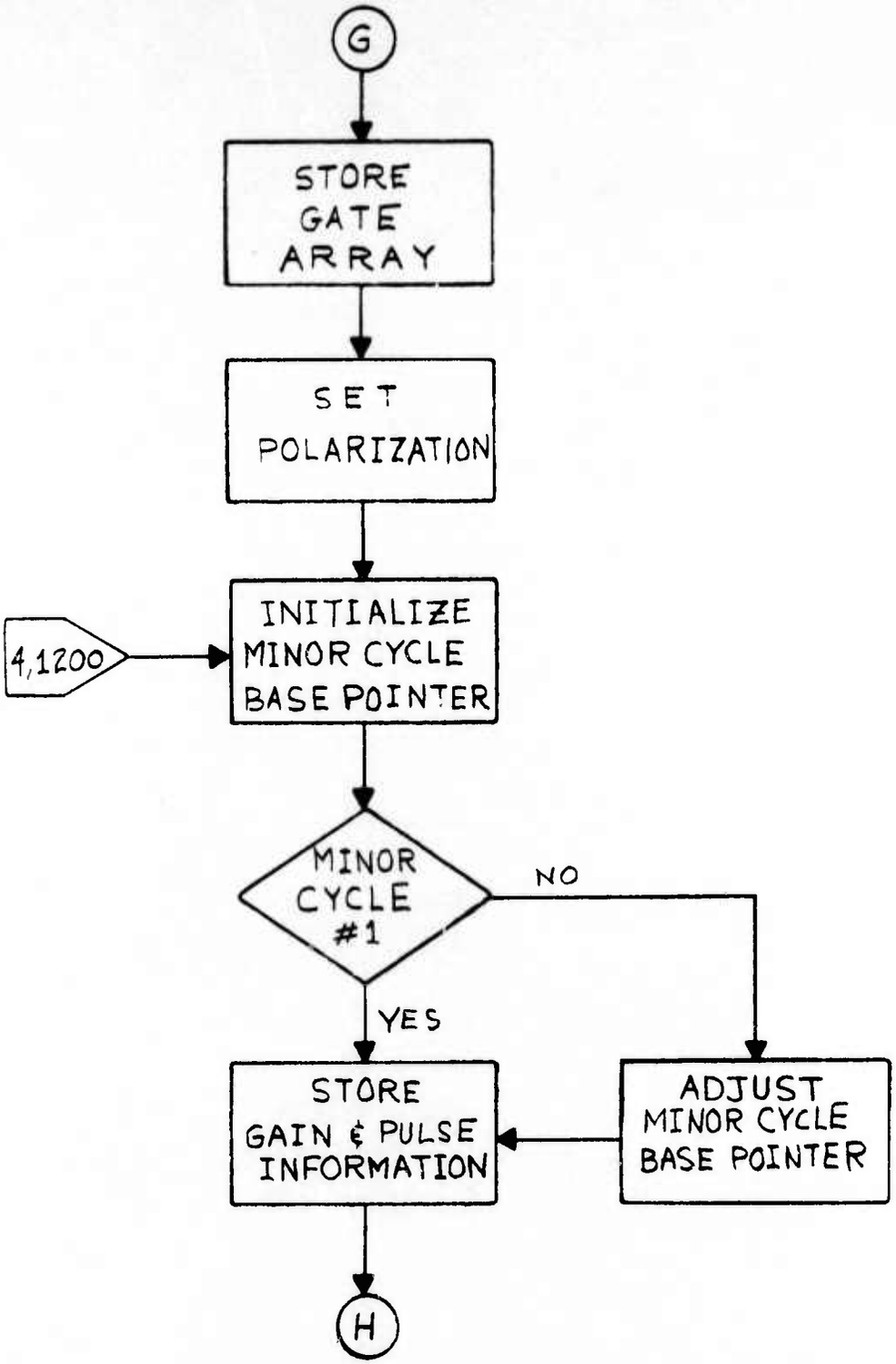

B
END OF FILE
YES
SET RETURN CODE = 44
RETURN
NO
2,180
DATA RECORD
NO
2, 120
YES
STORE PREVIOUS GMTIME
STORE HEADER DATA
GMTUPK
UNPACK GMTIME
C

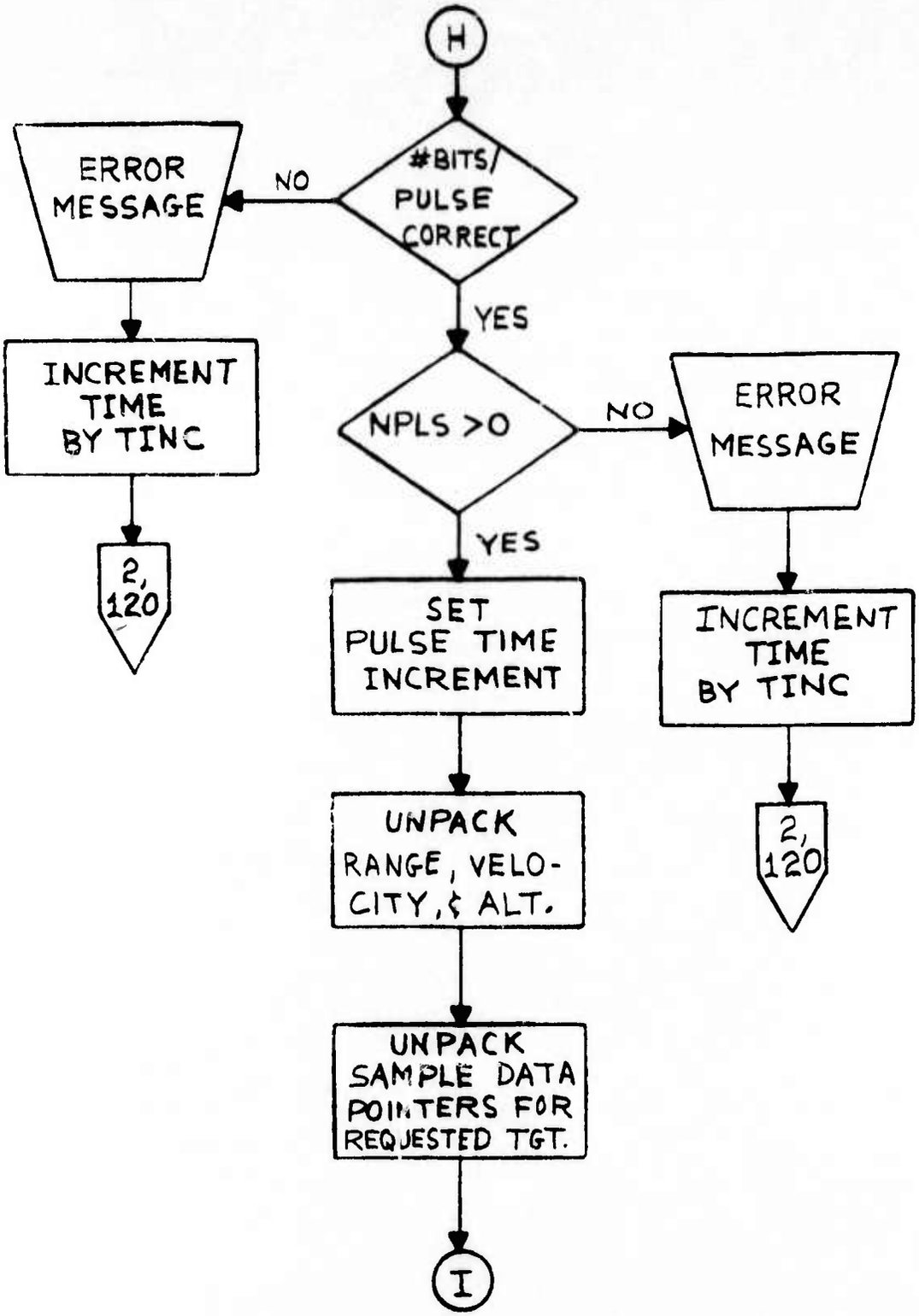

C
TSTART WITHIN CURRENT RECORD
NO
YES
VALID RECORD
NO
YES
CONTINUATION RECORD
NO
YES
TARGET ERROR
YES
2,120
NO
SET CONTINUE FLAG
8,1200
SET CHANNEL POINTER
D

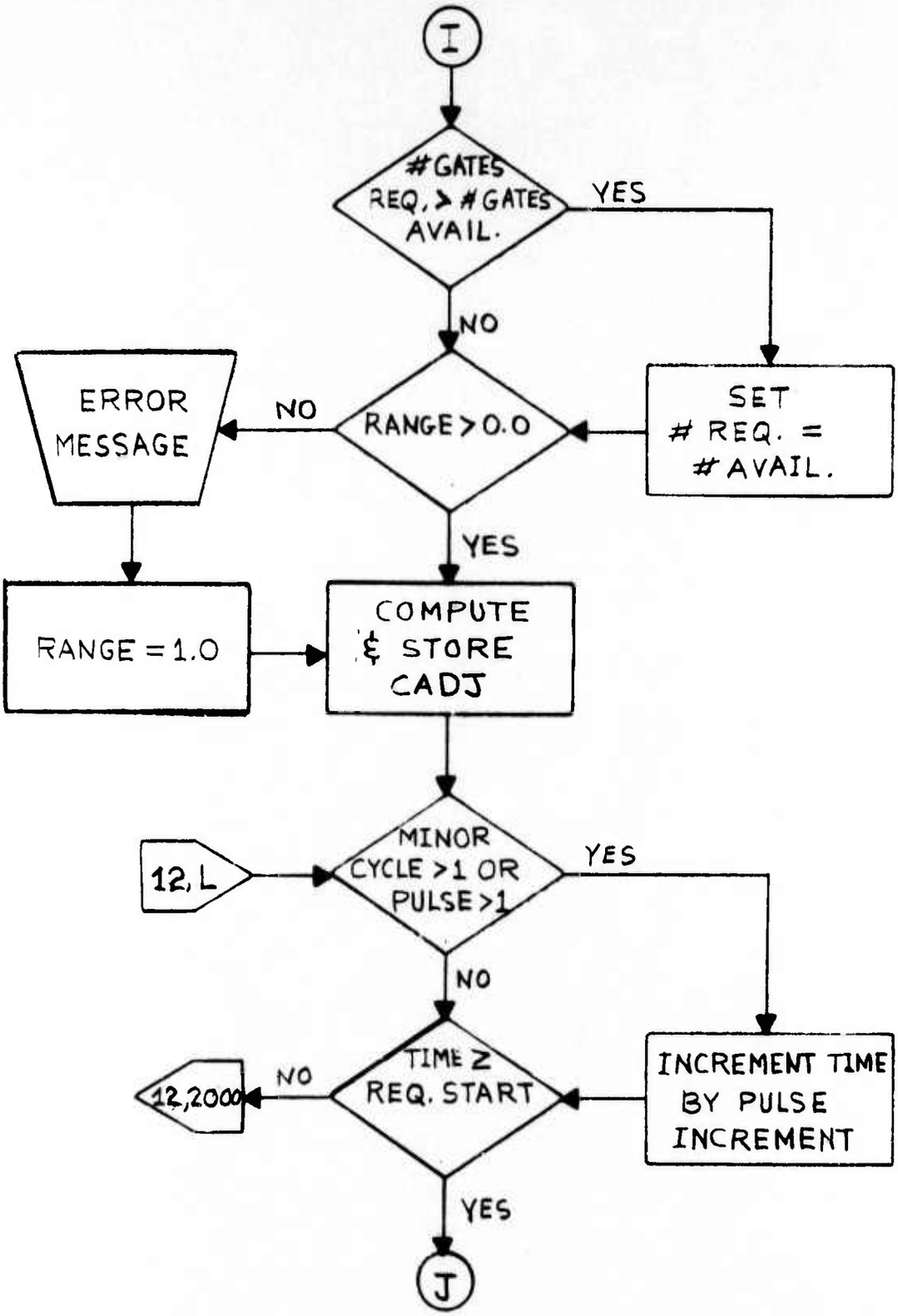

D
SET FREQUENCY
VITAL AMP & PHASE FORMATS PRESENT
NO
YES
SAVE CURRENT TIME AS TIMOLD; FINTM
SET RETURN CODE = 66
RETURN
ZERO NSAMPT
COMPUTE PRF & PRI
E

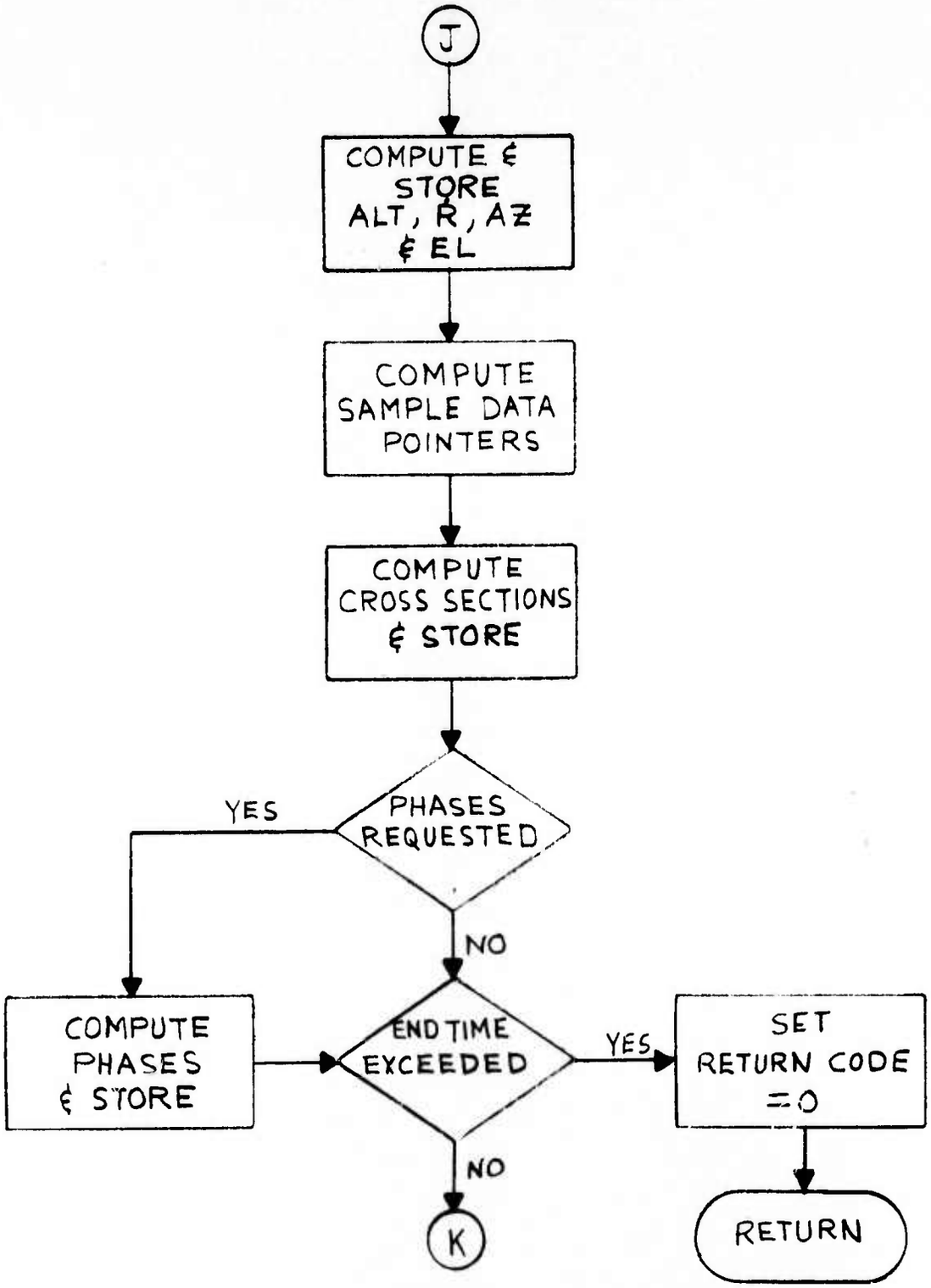

E
RADAR DATA PRESENT
NO
6,600
YES
UNPACK & STORE RADAR DATA
COMPUTE PORTION OF CADJ
6,600
VALID TARGETS CODED ON TAPE
NO
2,120
YES
STORE TARGET #s & PRIORITIES
F

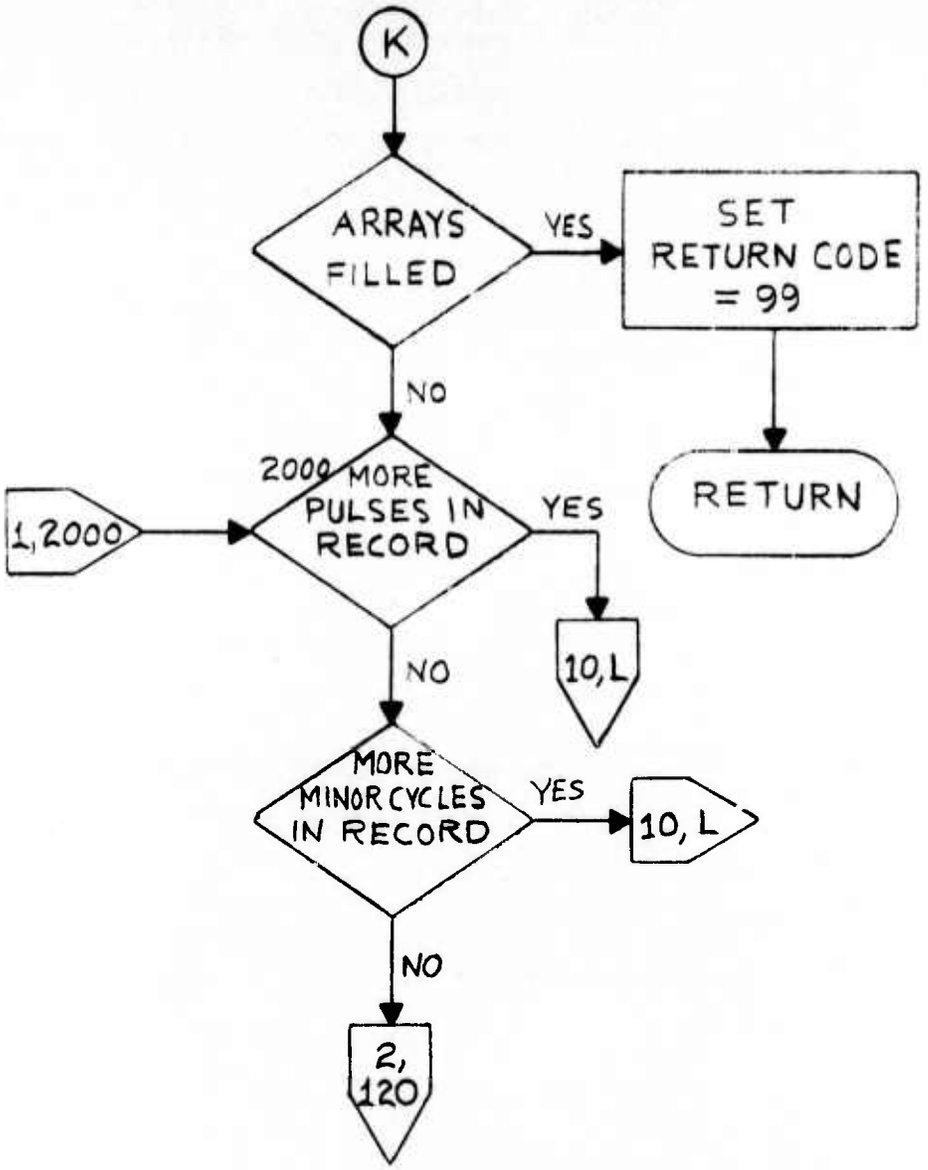

F
STORE MODE SPACING, # SAMPLES
COMPUTE DATA LOCATIONS ON TAPE RECORD
STORE STARTING GATES
REQUESTED TARGET PRESENT
NO
ERROR MESSAGE
YES
COMPUTE # BITS / PULSE
STOP
NO
POLARIZATION & SAMPLING PATTERNS VALID
YES
2,120
YES
BITS/PULSE VALID
NO
G
STOP


G
STORE GATE ARRAY
SET POLARIZATION
4,1200
INITIALIZE MINOR CYCLE BASE POINTER
MINOR CYCLE #1
NO
YES
STORE GAIN & PULSE INFORMATION
ADJUST MINOR CYCLE BASE POINTER
H


H
#BITS/ PULSE CORRECT
NO
ERROR MESSAGE
YES
INCREMENT TIME BY TINC
2, 120
NPLS >0
NO
ERROR MESSAGE
YES
SET PULSE TIME INCREMENT
INCREMENT TIME BY TINC
UNPACK RANGE, VELO-CITY, & ALT.
2, 120
UNPACK SAMPLE DATA POINTERS FOR REQUESTED TGT.
I


I
#GATES REQ. > #GATES AVAIL.
YES
NO
ERROR MESSAGE
NO
RANGE > 0.0
SET # REQ. = # AVAIL.
YES
RANGE = 1.0
COMPUTE & STORE CADJ
12, L
MINOR CYCLE > 1 OR PULSE > 1
YES
NO
12,2000
NO
TIME ≥ REQ. START
INCREMENT TIME BY PULSE INCREMENT
YES
J


J
COMPUTE & STORE ALT, R, AZ & EL
COMPUTE SAMPLE DATA POINTERS
COMPUTE CROSS SECTIONS & STORE
PHASES REQUESTED
YES
NO
COMPUTE PHASES & STORE
END TIME EXCEEDED
YES
NO
SET RETURN CODE =0
K
RETURN


K
ARRAYS FILLED
YES
SET RETURN CODE = 99
RETURN
NO
2000
1,2000
MORE PULSES IN RECORD
YES
10,L
NO
MORE MINOR CYCLES IN RECORD
YES
10, L
NO
2, 120

APPENDIX C

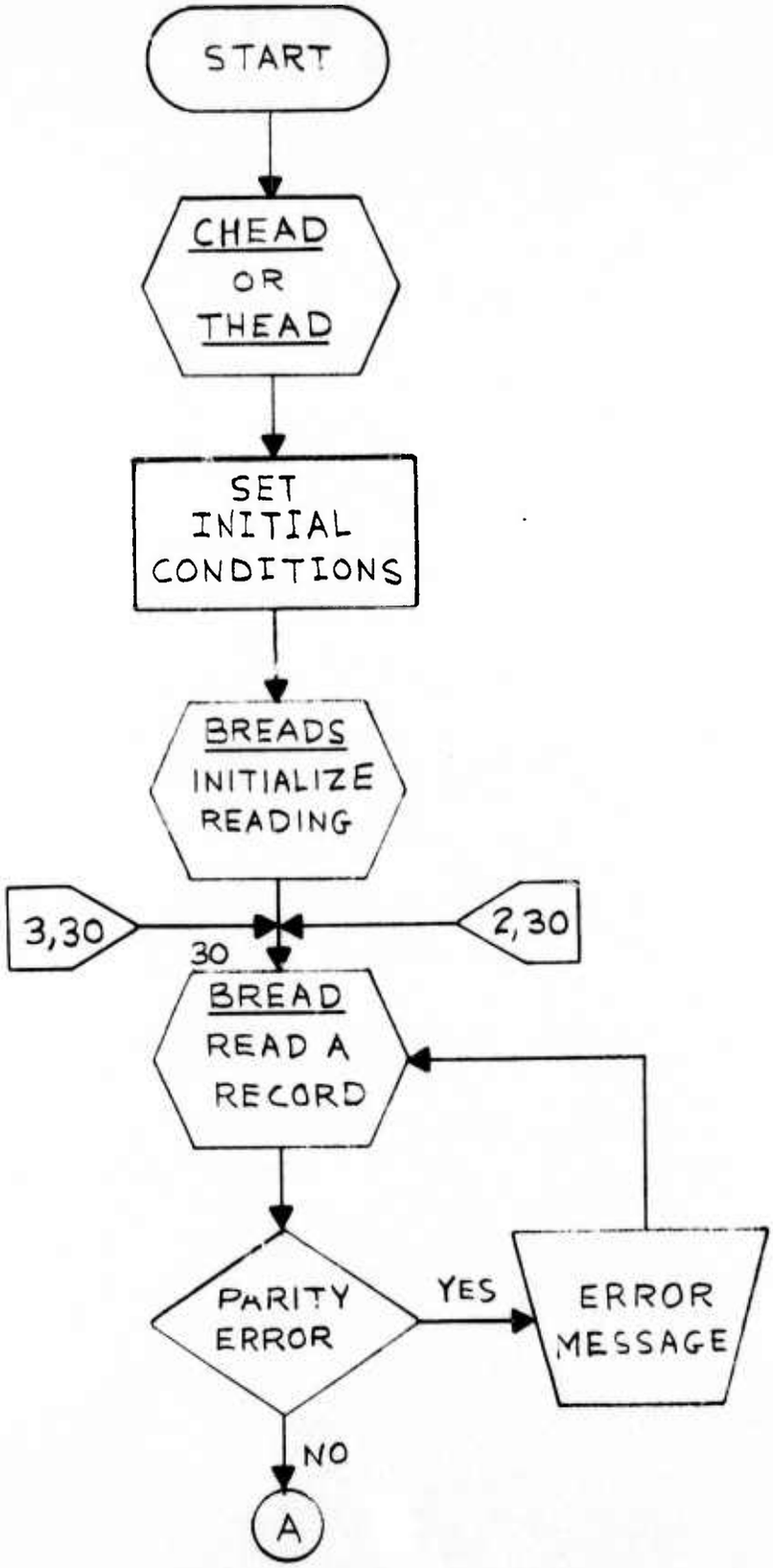
SUBROUTINE THEAD PROGRAM LISTING

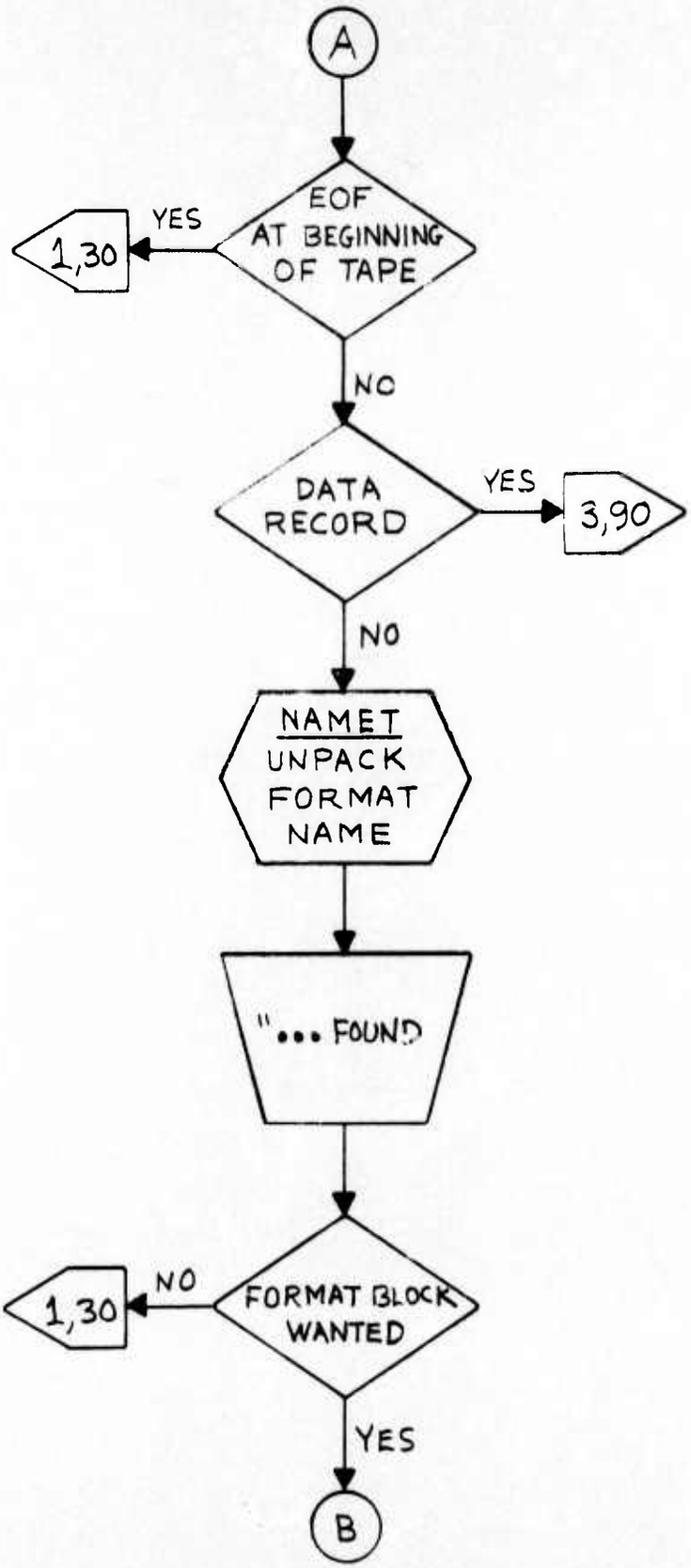
```
      SUBROUTINE THEAD(NEWPAS,*)        VERSION 3/15/71
      INTEGER*2 ITEM
      COMMON/TREAD/LN,IFLG,IADD,ICATFM(24),NAME(25),NI(24),IX(24),
     1TAMP(128,6),TPH(128,6),ITEM(2000)
      DIMENSION IBUF(2048,2),IBUF1(2048),IBUF2(2048),IDARR(8),NAMEX(25),
     1TEM(1)
      EQUIVALENCE (IBUF1(1),IBUF(1,1)),(IBUF2(1),IBUF(1,2))
      EQUIVALENCE(ITEM(1),TEM(1))
      DATA NAMEX/'TRHD','TRMA','TRTG','TRMI','TRSP','XSEC','ARSG','RR11'
     1,'GLOT','CHAF','BSMC','ASLP','AMP1','AMP2','AMP3','AMP4','AMP5',
     2'AMP6','PHA1','PHA2','PHA3','PHA4','PHA5','PHA6','HDRD'/
      DATA MAX/8192/,IT/1/
      DATA ICMS/'CMS '/
      IF(NEWPAS.NE.0)GO TO 40
      CALL WHICHV(ID)
      IOUT=6
      IF(ID.EQ.ICMS)IOUT=8
      NREC=0
      LN=1
      DO 20 I=1,24
      ICATFM(I)=0
      NAME(I)=NAMEX(I)
      NI(I)=0
      IX(I)=0
20    CONTINUE
      ICATFM(10)=1
      NAME(25)=NAMEX(25)
40    CALL BREADS(LN,IBUF1,IBUF2,MAX,IFL,INDX,LEN,IFLG,IADD)
60    CALL BREAD(LN)
      IF(NEWPAS.NE.0)RETURN
      NREC=NREC+1
      IF(IFLG.EQ.2)GO TO 380
      IF(IFLG.EQ.3.AND.IT.EQ.1)GO TO 60
      IF(IFLG.EQ.3)GO TO 480
      IF((IBUF(1,INDX)/16777216).EQ.2)GO TO 480
70    CALL NAMET(IADD,NAMED)
      IF(NAMED.EQ.NAMEX(25))GO TO 100
      WRITE(IOUT,80)NAMED
80    FORMAT(1X,A4,'  FOUND')
100   DO 120 I=1,25
      IF(NAMED.EQ.NAMEX(I))GO TO 140
120   CONTINUE
      GO TO 60
140   IF(NAMED.EQ.NAMEX(10))GO TO 60
      CALL FORM(IADD,ITEM(IT),IB,NAMED,NTEM,&260)
      IF(NAMED.EQ.NAMEX(25))GO TO 420
      DO 160 I=1,12
      IF(NAME(I).EQ.NAMED)GO TO 220
160   CONTINUE
      DO 180 J=13,18
      I=J-12
      IF(NAME(J).EQ.NAMED)GO TO 300
180   CONTINUE
      DO 200 J=19,24
      I=J-18
```

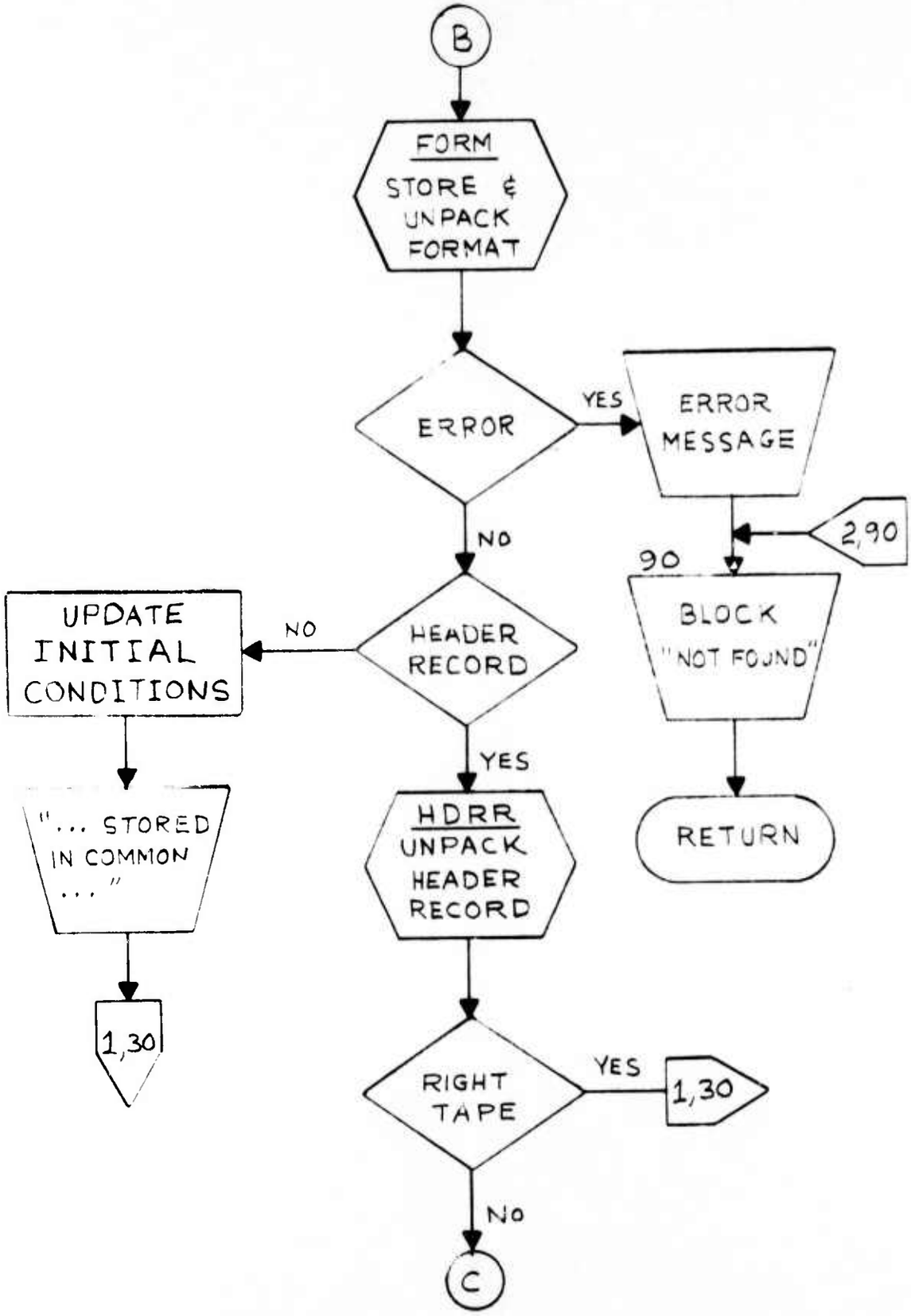
```
      IF(NAME(J).EQ.NAMED)GO TO 340
200   CONTINUE
      GO TO 60
220   ICATFM(I)=IB
      WRITE(IOUT,240)NAMED,NTEM
240   FORMAT(' FORMAT=',A4,'  STORED IN COMMON  NTEM=',I4)
      IX(I)=IT
      NI(I)=NTEM
      IT=IT+6*NTEM
      GO TO 60
260   WRITE(6,280)NAMED,NTEM
280   FORMAT(' NAME = 'A4,'   NTEM ='I5,' *ERROR* FORMAT TABLE LIMITED T
     10 700 ITEMS OR FORMAT TABLE HAS 0 LENGTH')
      GO TO 480
300   JT=IT
      ICATFM(J)=IB
      DO 320 J=1,127
      JT=JT+6
      TAMP(J,I)=TEM((JT+5)/2)
320   CONTINUE
      GO TO 60
340   JT=IT
      ICATFM(J)=IB
      DO 360 J=1,128
      JT=JT+6
      TPH(J,I)=TEM((JT+5)/2)
360   CONTINUE
      GO TO 60
380   WRITE(IOUT,400)NREC
400   FORMAT(' PARITY ERROR READING FORMAT RECORD',I6)
      GO TO 60
420   CALL HDRR(IADD,ITYP,IDARR)
      WRITE(IOUT,440)ITYP
440   FORMAT(' TYPE ',I2)
      IF((ITYP.EQ.1).OR.(ITYP.EQ.2))GO TO 60
      WRITE(6,460)
460   FORMAT(' TAPE NOT TRANSCRIPTION TAPE JOB TERMINATED BY THEAD')
      RETURN 1
480   DO 520 I=1,24
      IF(ICATFM(I).EQ.0)WRITE(IOUT,500)NAMEX(I)
500   FORMAT(1X,A4,' NOT FOUND')
520   CONTINUE
      WRITE(6,540)
540   FORMAT(' THEAD COMPIETE')
      RETURN
      END
```

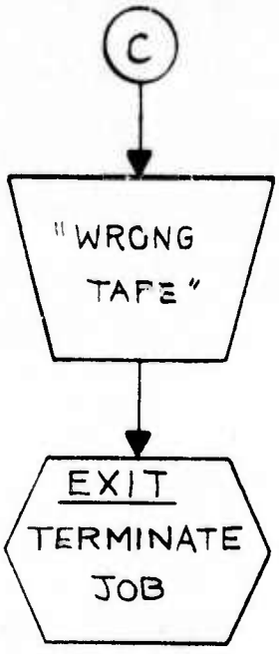
APPENDIX D

SUBROUTINE THEAD FLOW DIAGRAM




A
EOF AT BEGINNING OF TAPE
YES
1,30
NO
DATA RECORD
YES
3,90
NO
NAMET
UNPACK FORMAT NAME
"... FOUND
FORMAT BLOCK WANTED
NO
1,30
YES
B


B
FORM
STORE & UNPACK FORMAT
ERROR
YES
ERROR MESSAGE
2,90
90
BLOCK "NOT FOUND"
RETURN
NO
HEADER RECORD
NO
UPDATE INITIAL CONDITIONS
"... STORED IN COMMON ..."
1,30
YES
HDRR
UNPACK HEADER RECORD
RIGHT TAPE
YES
1,30
NO
C


C
"WRONG TAPE"
EXIT
TERMINATE JOB

APPENDIX E
SUBROUTINE THEAD OUTPUT

```
YURL   FOUND
GLOT   FOUND
FORMAT=GLOT   STORED IN COMMON   NTEM=    7
TRHD   FOUND
FORMAT=TRHD   STORED IN COMMON   NTEM=   15
TRMA   FOUND
FORMAT=TRMA   STORED IN COMMON   NTEM=    7
TRTG   FOUND
FORMAT=TRTG   STORED IN COMMON   NTEM=   14
TRMI   FOUND
FORMAT=TRMI   STORED IN COMMON   NTEM=    9
TRSP   FOUND
FORMAT=TRSP   STORED IN COMMON   NTEM=    2
AMP1   FOUND
AMP2   FOUND
AMP3   FOUND
AMP4   FOUND
AMP5   FOUND
AMP6   FOUND
RDID   FOUND
BSMC   FOUND
FORMAT=BSMC   STORED IN COMMON   NTEM=   28
AACC   FOUND
ASLP   FOUND
FORMAT=ASLP   STORED IN COMMON   NTEM=    3
PHA1   FOUND
PHA2   FOUND
PHA3   FOUND
PHA4   FOUND
PHA5   FOUND
PHA6   FOUND
CHAF   FOUND
BCAL   FOUND
XSEC   FOUND
FORMAT=XSEC   STORED IN COMMON   NTEM=   19
SCAN   FOUND
ARSG NOT FOUND
RR11 NOT FOUND
THEAD COMPLETE
```

APPENDIX F

SUBROUTINE GMTUPK PROGRAM LISTING

```
      SUBROUTINE GMTUPK(TIME)
      DIMENSION SECMUL(6,2),T(2),LIM(2) ,ISHIFT(6,2)
      COMMON/TIMCOM/ITM(2)
      DOUBLE PRECISION TIME,T,TSUM
      DATA SECMUL/36000.0,3600.0,600.0,60.0,10.0,1.0,0.1,0.01,0.001,0.0,
     10.0,0.0/
      DATA ISHIFT/1048576,65536,4096,256,16,1,256,16,1,1,1,1/
      DATA LIM/6,3/
      DO 200 J=1,2
      TSUM=0.0
      IDIG=ITM(J)
      ILIM=LIM(J)
      DO 100 I=1,ILIM
      NSLIDE=IDIG/ISHIFT(I,J)
      TSUM=TSUM+FLOAT(NSLIDE)*SECMUL(I,J)
      IDIG=IDIG-NSLIDE*ISHIFT(I,J)
100   CONTINUE
      T(J)=TSUM
200   CONTINUE
      TIME=T(1)+T(2)
      RETURN
      END
```